

# **Отчёт по лабораторной работе №4**

**дисциплина: Архитектура компьютера**

Клименко Алёна Сергеевна

# Содержание

<b>1</b>	<b>Цель работы</b>	<b>5</b>
<b>2</b>	<b>Задание</b>	<b>6</b>
<b>3</b>	<b>Выполнение лабораторной работы</b>	<b>9</b>
3.1	Программа Hello world! . . . . .	9
3.2	Транслятор NASM . . . . .	10
3.3	Расширенный синтаксис командной строки NASM . . . . .	11
3.4	Компоновщик LD . . . . .	11
3.5	Запуск исполняемого файла . . . . .	11
3.6	Задания для самостоятельной работы . . . . .	12
<b>4</b>	<b>Выводы</b>	<b>14</b>
<b>5</b>	<b>Список литературы</b>	<b>15</b>

# Список иллюстраций

3.1	Создание рабочей директории . . . . .	9
3.2	Создание .asm файла . . . . .	9
3.3	Редактирование файла . . . . .	10
3.4	Компиляция программы . . . . .	10
3.5	Возможности синтаксиса NASM . . . . .	11
3.6	Отправка файла компоновщику . . . . .	11
3.7	Создание исполняемого файла . . . . .	11
3.8	Запуск программы . . . . .	12
3.9	Создание копии . . . . .	12
3.10	Проверка работоспособности скомпонованной программы . . . . .	12
3.11	Отправка файлов в локальный репозиторий . . . . .	12
3.12	Загрузка изменений . . . . .	13

## **Список таблиц**

# 1 Цель работы

Цель данной лабораторной работы - освоить процедуры компиляции и сборки программ, написанных на ассемблере NASM.

## 2 Задание

1. Создание программы Hello world!
2. Работа с транслятором NASM
3. Работа с расширенным синтаксисом командной строки NASM
4. Работа с компоновщиком LD
5. Запуск исполняемого файла
6. Выполнение заданий для самостоятельной работы.

Основными функциональными элементами любой ЭВМ являются центральный процессор, память и периферийные устройства. Взаимодействие этих устройств осуществляется через общую шину, к которой они подключены. Физически шина представляет собой большое количество проводников, соединяющих устройства друг с другом. В современных компьютерах проводники выполнены в виде электропроводящих дорожек на материнской плате. Основной задачей процессора является обработка информации, а также организация координации всех узлов компьютера. В состав центрального процессора входят следующие устройства: - арифметико-логическое устройство (АЛУ) — выполняет логические и арифметические действия, необходимые для обработки информации, хранящейся в памяти; - устройство управления (УУ) — обеспечивает управление и контроль всех устройств компьютера; - регистры — сверхбыстрая оперативная память небольшого объёма, входящая в состав процессора, для временного хранения промежуточных результатов выполнения инструкций; регистры процессора делятся на два типа: регистры общего назначения и специальные регистры. Для того, чтобы писать программы на

ассемблере, необходимо знать, какие регистры процессора существуют и как их можно использовать. Большинство команд в программах написанных на ассемблере используют регистры в качестве операндов. Практически все команды представляют собой преобразование данных хранящихся в регистрах процессора, это например пересылка данных между регистрами или между регистрами и памятью, преобразование (арифметические или логические операции) данных хранящихся в регистрах. Доступ к регистрам осуществляется не по адресам, как к основной памяти, а по именам. Каждый регистр процессора архитектуры x86 имеет свое название, состоящее из 2 или 3 букв латинского алфавита. В качестве примера приведем названия основных регистров общего назначения (именно эти регистры чаще всего используются при написании программ): - RAX, RCX, RDX, RBX, RSI, RDI — 64-битные - EAX, ECX, EDX, EBX, ESI, EDI — 32-битные - AX, CX, DX, BX, SI, DI — 16-битные - AH, AL, CH, CL, DH, DL, BH, BL — 8-битные

Другим важным узлом ЭВМ является оперативное запоминающее устройство (ОЗУ). ОЗУ — это быстродействующее энергозависимое запоминающее устройство, которое напрямую взаимодействует с узлами процессора, предназначенное для хранения программ и данных, с которыми процессор непосредственно работает в текущий момент. ОЗУ состоит из одинаковых пронумерованных ячеек памяти. Номер ячейки памяти — это адрес хранящихся в ней данных. Периферийные устройства в составе ЭВМ: - устройства внешней памяти, которые предназначены для долговременного хранения больших объёмов данных. - устройства ввода-вывода, которые обеспечивают взаимодействие ЦП с внешней средой.

В основе вычислительного процесса ЭВМ лежит принцип программного управления. Это означает, что компьютер решает поставленную задачу как последовательность действий, записанных в виде программы.

Коды команд представляют собой многоразрядные двоичные комбинации из 0 и 1. В коде машинной команды можно выделить две части: операционную и адресную. В операционной части хранится код команды, которую необходимо

выполнить. В адресной части хранятся данные или адреса данных, которые участвуют в выполнении данной операции. При выполнении каждой команды процессор выполняет определённую последовательность стандартных действий, которая называется командным циклом процессора. Он заключается в следующем: 1. формирование адреса в памяти очередной команды; 2. считывание кода команды из памяти и её дешифрация; 3. выполнение команды; 4. переход к следующей команде.

Язык ассемблера (assembly language, сокращённо asm) — машинно-ориентированный язык низкого уровня. NASM — это открытый проект ассемблера, версии которого доступны под различные операционные системы и который позволяет получать объектные файлы для этих систем. В NASM используется Intel-синтаксис и поддерживаются инструкции x86-64.



## 3 Выполнение лабораторной работы

### 3.1 Программа Hello world!

В домашней директории создаю каталог, в котором буду хранить файлы для этой лабораторной работы. (рис. 3.1)

```
klimenkoalena@fedora:~/work/study/2024-2025/Архитектура компьютера/arch-pc/labs/lab02/report$ cd ~/
klimenkoalena@fedora:~$ mkdir -p ~/work/arch-pc/lab04
klimenkoalena@fedora:~$ cd ~/work/
klimenkoalena@fedora:~/work$ cd ~/work/arch-pc/lab04/
```

Рис. 3.1: Создание рабочей директрории

Далее создаю файл hello.asm, в котором буду писать программу на языке ассемблера. (рис. 3.2)

```
klimenkoalena@fedora:~$ mkdir -p ~/work/arch-pc/lab04
klimenkoalena@fedora:~$ cd ~/work/
klimenkoalena@fedora:~/work$ cd ~/work/arch-pc/lab04/
klimenkoalena@fedora:~/work/arch-pc/lab04$ touch hello.asm
klimenkoalena@fedora:~/work/arch-pc/lab04$ mousepad hello.asm
```

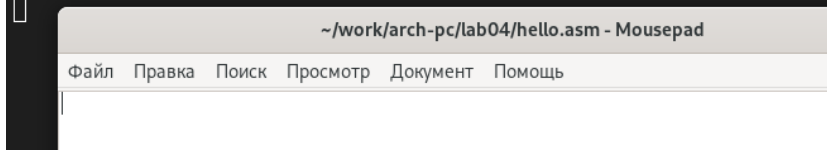
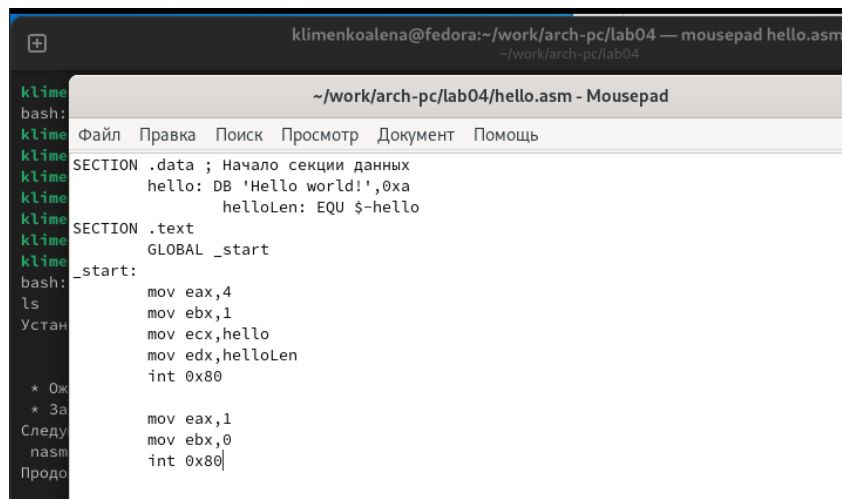


Рис. 3.2: Создание .asm файла

С помощью редактора mousepad пишу программу в созданном файле. (рис. 3.3)

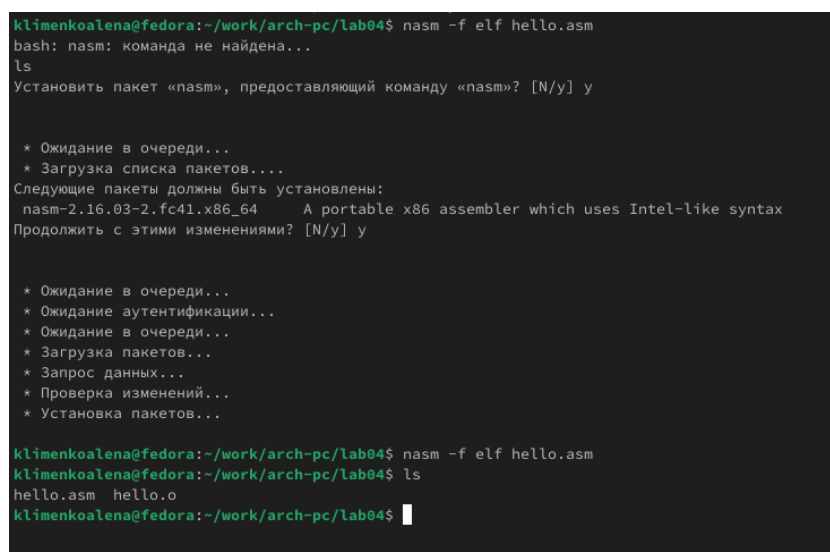


```
klime klimenkoalena@fedora:~/work/arch-pc/lab04 — mousepad hello.asm
~/work/arch-pc/lab04
klime ~ /work/arch-pc/lab04/hello.asm - Mousepad
klime Файл Правка Поиск Просмотр Документ Помощь
klime SECTION .data ; Начало секции данных
klime     hello: DB 'Hello world!',0xa
klime           helloLen: EQU $-hello
klime SECTION .text
klime     GLOBAL _start
bash: _start:
ls      mov eax,4
Устан   mov ebx,1
        mov ecx,hello
        mov edx,helloLen
        int 0x80
* Ож
* За
Следу   mov eax,1
nasm    mov ebx,0
Продо   int 0x80
```

Рис. 3.3: Редактирование файла

## 3.2 Транслятор NASM

Компилирую с помощью NASM свою программу. (рис. 3.4)



```
klimenkoalena@fedora:~/work/arch-pc/lab04$ nasm -f elf hello.asm
bash: nasm: команда не найдена...
ls
Установить пакет «nasm», предоставляющий команду «nasm»? [N/y] y

* Ожидание в очереди...
* Загрузка списка пакетов...
Следующие пакеты должны быть установлены:
nasm-2.16.03-2.fc41.x86_64      A portable x86 assembler which uses Intel-like syntax
Продолжить с этими изменениями? [N/y] y

* Ожидание в очереди...
* Ожидание аутентификации...
* Ожидание в очереди...
* Загрузка пакетов...
* Запрос данных...
* Проверка изменений...
* Установка пакетов...

klimenkoalena@fedora:~/work/arch-pc/lab04$ nasm -f elf hello.asm
klimenkoalena@fedora:~/work/arch-pc/lab04$ ls
hello.asm hello.o
klimenkoalena@fedora:~/work/arch-pc/lab04$
```

Рис. 3.4: Компиляция программы

### 3.3 Расширенный синтаксис командной строки NASM

Выполняю команду, указанную на (рис. 3.5), компилируется исходный файл hello.asm в obj.o, расширение .o говорит о том, что файл - объектный, помимо него флаги -g -l подготовят файл отладки и листинга соответственно.

```
klimenkoalena@fedora:~/work/arch-pc/lab04$ nasm -o obl.o -f elf -g -l list.lst hello.asm
klimenkoalena@fedora:~/work/arch-pc/lab04$ ls
hello.asm  hello.o  list.lst  obl.o
klimenkoalena@fedora:~/work/arch-pc/lab04$
```

Рис. 3.5: Возможности синтаксиса NASM

### 3.4 Компоновщик LD

Далее передаю объектный файл компоновщику, с помощью команды ld. (рис. 3.6)

```
klimenkoalena@fedora:~/work/arch-pc/lab04$ ld -m elf_i386 hello.o -o hello
klimenkoalena@fedora:~/work/arch-pc/lab04$ ls
hello  hello.asm  hello.o  list.lst  obl.o
klimenkoalena@fedora:~/work/arch-pc/lab04$
```

Рис. 3.6: Отправка файла компоновщику

Выполняю следующую команду, результатом будет созданный файл main, скомпонованный из объектного файла obl.o. (рис. 3.7)

```
klimenkoalena@fedora:~/work/arch-pc/lab04$ ld -m elf_i386 obl.o -o main
klimenkoalena@fedora:~/work/arch-pc/lab04$ ls
hello  hello.asm  hello.o  list.lst  main  obl.o
klimenkoalena@fedora:~/work/arch-pc/lab04$
```

Рис. 3.7: Создание исполняемого файла

### 3.5 Запуск исполняемого файла

Запускаю исполняемый файл для проверки. (рис. 3.8)

```

klimenkoalena@fedora:~/work/arch-pc/lab04$ ./hello
Hello world!
klimenkoalena@fedora:~/work/arch-pc/lab04$

```

Рис. 3.8: Запуск программы

## 3.6 Задания для самостоятельной работы

Создаю копию файла для последующей работы с ней. (рис. 3.9)

```

klimenkoalena@fedora:~/work/arch-pc/lab04$ cp hello.asm lab4.asm
klimenkoalena@fedora:~/work/arch-pc/lab04$ ls
hello  hello.asm  hello.o  lab4.asm  list.lst  main  obl.o
klimenkoalena@fedora:~/work/arch-pc/lab04$

```

Рис. 3.9: Создание копии

Редактирую копию файла, поменяв 'Hello, world!' на свое имя и фамилию Транс-  
лирую копию файла в объектный файл, компоную и запускаю. (рис. 3.10)

```

klimenkoalena@fedora:~/work/arch-pc/lab04$ nasm -f elf lab4.asm
klimenkoalena@fedora:~/work/arch-pc/lab04$ ld -m elf_i386 lab4.o lab4
ld: невозможно найти lab4: Нет такого файла или каталога
klimenkoalena@fedora:~/work/arch-pc/lab04$ ls
hello  hello.asm  hello.o  lab4.asm  lab4.o  list.lst  main  obl.o
klimenkoalena@fedora:~/work/arch-pc/lab04$ ld -m elf_i386 lab4.o -o lab4
klimenkoalena@fedora:~/work/arch-pc/lab04$ ls
hello  hello.asm  hello.o  lab4  lab4.asm  lab4.o  list.lst  main  obl.o
klimenkoalena@fedora:~/work/arch-pc/lab04$ ./lab4
Klivenko Alena
klimenkoalena@fedora:~/work/arch-pc/lab04$

```

Рис. 3.10: Проверка работоспособности скомпонованной программы

Убедившись в корректности работы программы, копирую рабочие файлы в  
свой локальный репозиторий. (рис. 3.11)

```

klimenkoalena@fedora:~/work/arch-pc/lab04$ cp hello.asm lab4.asm ../../study/2024-2025/Архитектура\ компьютера/arch-p
c/labs/lab04/
klimenkoalena@fedora:~/work/arch-pc/lab04$ cd ../../study/2024-2025/Архитектура\ компьютера/arch-pc/labs/lab04/
bash: cd: ../../study/2024-2025/Архитектура\ компьютера/arch-pc/labs/lab04/: Нет такого файла или каталога
klimenkoalena@fedora:~/work/arch-pc/lab04$ cd ../../study/2024-2025/Архитектура\ компьютера/arch-pc/labs/lab04/
klimenkoalena@fedora:~/work/study/2024-2025/Архитектура\ компьютера/arch-pc/labs/lab04$ ls
hello.asm  lab4.asm  presentation  report
klimenkoalena@fedora:~/work/study/2024-2025/Архитектура\ компьютера/arch-pc/labs/lab04$

```

Рис. 3.11: Отправка файлов в локальный репозиторий

Загрузка изменений на свой репозиторий на GitHub. (рис. 3.12)

```

klimenkoalena@fedora:~/work/study/2024-2025/Архитектура компьютера/arch-pc/labs/lab04$ git add .
klimenkoalena@fedora:~/work/study/2024-2025/Архитектура компьютера/arch-pc/labs/lab04$ git status
Текущая ветка: master
Эта ветка соответствует «origin/master».

Изменения, которые будут включены в коммит:
(используйте «git restore --staged <файл>...», чтобы убрать из индекса)
    новый файл:    hello.asm
    новый файл:    lab4.asm

Неотслеживаемые файлы:
(используйте «git add <файл>...», чтобы добавить в то, что будет включено в коммит)
    ../lab03/report/image/image1ab03.zip

klimenkoalena@fedora:~/work/study/2024-2025/Архитектура компьютера/arch-pc/labs/lab04$ git commit -m "feat(main): upload 4 lab work"
[master e15c6cc] feat(main): upload 4 lab work
 2 files changed, 38 insertions(+)
 create mode 100644 labs/lab04/hello.asm
 create mode 100644 labs/lab04/lab4.asm
klimenkoalena@fedora:~/work/study/2024-2025/Архитектура компьютера/arch-pc/labs/lab04$ git push
Перечисление объектов: 9, готово.
Подсчет объектов: 100% (3/3), готово.
Сжатие объектов: 100% (6/6), готово.
Запись объектов: 100% (6/6), 656 байтов | 164.00 КиБ/с, готово.
Total 6 (delta 3), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (3/3), completed with 2 local objects.
To github.com:Alstrr/study_2024-2025_arh-pc.git
 33e1781..e15c6cc master -> master
klimenkoalena@fedora:~/work/study/2024-2025/Архитектура компьютера/arch-pc/labs/lab04$

```

Рис. 3.12: Загрузка изменений

## 4 Выводы

При выполнении данной лабораторной работы я освоила процедуры компиляции и сборки программ, написанных на ассемблере NASM.

## **5 Список литературы**

1. Курс на ТУИС
2. Лабораторная работа №4